OBJECT ORIENTED PROGRAMMING WITH JAVA 8– LAB 15

1. Create Product class with data members id, name, category and price and write two constructors, one default and other parametrized.

Create a list of Products and do the following operations using Stream.

a) Get a list of products which belongs to category “Books” with price > 100.

b) Get the total no: of products.

c) Find the total price of all products.

Ans=

Code-Part1,

```
import java.util.ArrayList;
import java.util.List;
import java.util.stream.Collectors;

class Product
{
        int id;
        String name;
        String category;
        double price;

        public Product()
        {
        }

        public Product(int id, String name, String category, double price)
        {
         this.id = id;
         this.name = name;
         this.category = category;
         this.price = price;
        }

        public int getId()
        {
         return id;
        }

        public String getName()
        {
         return name;
        }

        public String getCategory()
        {
         return category;
        }

        public double getPrice()
        {
         return price;
        }
}
```

Part2,

```
public class BooksMain
{
        public static void main(String[] args)
        {
         List<Product> productList = new ArrayList<>();
         productList.add(new Product(1, "Yayati", "Books", 150.0));
         productList.add(new Product(2, "Mritunjaya", "Books", 120.0));
         productList.add(new Product(3, "Phone 1", "Electronics", 500.0));
         productList.add(new Product(4, "Shirt 1", "Clothing", 25.0));

         List<Product> expensiveBooks = productList.stream()
                .filter(product -> product.getCategory().equals("Books") && product.getPrice() > 100)
                .collect(Collectors.toList());

         long totalProducts = productList.stream().count();

         double totalPrice = productList.stream()
                .mapToDouble(Product::getPrice)
                .sum();

         System.out.println("Expensive Books:");
         expensiveBooks.forEach(product -> System.out.println(product.getName()));

         System.out.println("Total Number of Products: " + totalProducts);
         System.out.println("Total Price of All Products: " + totalPrice);
        }
}
```

Execution-

```
C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>javac BooksMain.java

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>java BooksMain
Expensive Books:
Yayati
Mritunjaya
Total Number of Products: 4
Total Price of All Products: 795.0

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>
```

2. Create an infinite stream of multiples of 5. Display the stream elements after skipping first 3 and limit to 10 elements.

Ans=

Code-

```
import java.util.stream.Stream;

public class InfiniteMultiplesOfFive
{
        public static void main(String[] args)
        {
         Stream<Integer> multiplesOfFive = Stream.iterate(5, n -> n + 5);

         multiplesOfFive
            .skip(3)
            .limit(10)
            .forEach(System.out::println);
        }
}
```

Execution-

```
C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>javac InfiniteMultiplesOfFive.java

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>java InfiniteMultiplesOfFive
20
25
30
35
40
45
50
55
60
65

C:\Users\p7pha\OneDrive\Desktop\Cdac DBDA\JAVA>
```